In [1]:

```
from network import NeuralNetwork
from layer import LinearLayer, Sigmoid, Relu, Softmax, Tanh, LeakyRelu
from loss import MSE, CrossEntropy
from optimizer import MBGD
from train import train

'''
Simple neural network / library written from scratch only using numpy.
Followed architecture of Joel Grus.

Check the python files in the aknet/ directory
to see the code I wrote for the layers, network, the loss function, the optimi
and everything else. I seperated everything out into modules
following Joel Gru's architecture. This made this neural network very modular
'''
%matplotlib inline
```

In [2]:

```
net = NeuralNetwork([
    LinearLayer(inputSize=64, outputSize=16),
    LeakyRelu(),
    LinearLayer(inputSize=16, outputSize=10),
    LeakyRelu(),
    Softmax()
])
```

In [3]:

```
import numpy as np
from sklearn.datasets import load_digits
digits = load_digits()
inputs = digits.data
for x in inputs:
    x /= 255
targets = []
for num in digits.target:
    baz = np.zeros(10)
    baz[num] = 1
    targets.append(baz)
targets = np.array(targets)
from sklearn.model_selection import train_test_split
inputs,xtest,targets, ytest = train_test_split(inputs, targets, test_size = 0
```

In [4]:

```
"Run cell train the Neural Network if you want"
train(net, inputs, targets, loss= CrossEntropy(), num_epochs=600, optimizer=MB
```

```
0 3782.0889145447477
1 3491.3633879351514
2 3415.2581266591137
3 3377.8573254809476
4 3354.98656117703
5 3338.74177546294
6 3327.570179332013
7 3319.0960367279104
8 3313.0877993034496
9 3308.173057565923
10 3305.1874939075815
11 3302.6420469317604
12 3300.5946643233265
13 3299.169670085772
14 3298.3748755529205
15 3297.2704977299586
16 3296.6458324363084
17 3295.888349313207
18 3295.6652992273316
19 3295.0287452150957
```

In [5]:

```
'''
We will just load a serialized version of the parameters,
to save time
'''
net.loadParamsFromFile("serializedMNIST.json")
```

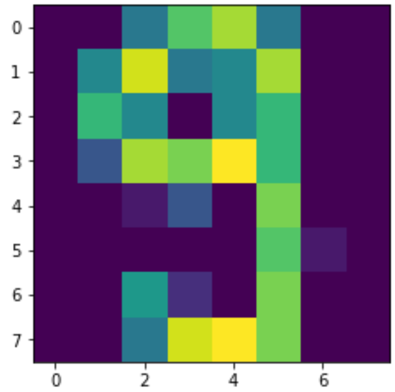
In [9]:

```
'''
Demonstrating NN predicting 10 images
it hasn't seen from the test set
'''

from matplotlib import pyplot as plt
import random
randidx = lambda: np.random.randint(low = 0, high = len(xtest))
for i in range(10):
    r = randidx()
    X, Y = xtest[r], ytest[r]
    plt.imshow(X.reshape(8,8))
    predicted = net.forward(X)
    print("Predicted:",np.argmax(predicted), "Actual:", np.argmax(Y),"\n\nImag
    plt.show()
```

```
Predicted: 9 Actual: 9

Image:
```



In [11]:

```
!
```

```
dataIterator.py  mnist.py               optimizer.py          tensor.py
__init__.py      network.py             __pycache__           train.py
layer.py         NeuralNetwork.ipynb    serialized.json       xor.py
loss.py          NeuralNetwork.py.ipynb serializedMNIST.json
```

In [ ]: